

Kenya Unit Trusts: Money Market Fund(KES) Analysis

Can Kenyan Money Market Funds gurantee capital preservation?


Charles Kabui


2024-03-03


Lorem Ipsum is simply dummy text of the printing and typesetting industry. Lorem Ipsum has been the industry’s standard dummy text ever since the 1500s, when an unknown printer took a galley of type and scrambled it to make a type specimen book. It has survived not only five centuries, but also the leap into electronic typesetting, remaining essentially unchanged. It was popularised in the 1960s with the release of Letraset sheets containing Lorem Ipsum passages, and more recently with desktop publishing software like Aldus PageMaker including versions of Lorem Ipsum.


Table of Contents


Introduction	2
Analysis	2
Imports	2
Fund Managers	3
Add Columns	4
Getting the Perormance	4
Cytton Research	5

 Open in Colab

 launch binder

 View On GitHub

 Launch in Deepnote

 Download as PDF

Introduction

In Kenya, a good number of the population have a bank account, even if not the traditional bank account. It is estimated that by year 2029, 99.9% of Kenyans will be banked,¹. With increase in financial access, financial literacy and regular individuals will want to venture in the territory of financial assets hunting for higher interest rates. One of the most attractive entry level high-yield financial asset is the unit trust, specifically the money market funds. Its easy to start, deposit, withdraw, and its interest is daily compounding, while offering higher interest than any bank.

Analysis

Imports

```
import sys
import os

# Add parent directory to sys.path
root_dir = os.path.abspath(os.path.join(os.getcwd(), '../..'))
sys.path.append(root_dir)

%load_ext autoreload
%autoreload 2
```

```
import pandas as pd
from pyppeteer.page import Page
import asyncio
import json
import io
from bs4 import BeautifulSoup, Tag
from urllib.request import urlopen
from pyppeteer.page import Request
from tqdm import tqdm
from python_utils.web_screenshot import web_screenshot_async
from python_utils.get_browser import get_browser_page_async
```

¹Population share with banking account in Kenya 2014-2029 Published by J. Degenhard, Jan 30, 2024.
<https://www.statista.com/forecasts/1149636/bank-account-penetration-forecast-in-kenya>

Fund Managers

Let's start of by listing all the certified fund managers in Kenya by CMA.²

Lets start with a screenshot of the webpage.

```
async def action(page: Page):
    await page.waitForSelector('ul.module-accordion')
    elements = await page.querySelectorAll('li.accordion-title')
    # Iterate through the elements to find the one containing 'FUND MANAGERS'
    for element in elements:
        text_content = await page.evaluate('(element) => element.textContent', element)
        if 'FUND MANAGERS' in text_content:
            # Click on the target element
            await element.click()
            break
    else:
        print('Element not found')
    await page.waitForSelector('li.current.builder-accordion-active')
    await asyncio.sleep(1)

await web_screenshot_async(
    "https://www.cma.or.ke/licensees-market-players/",
    action = action,
    width=1500)
```

Let's query the "Fund Managers" table.

```
url_response = urlopen("https://www.cma.or.ke/licensees-market-players/").read()
fund_managers_html_table = BeautifulSoup(url_response, "html.parser")\
    .find('span', string="FUND MANAGERS")\
        .parent\
            .parent\
                .parent\
                    .find('table')

fund_managers_df = pd.read_html(io.StringIO(str(fund_managers_html_table)))[0].dropna()
fund_managers_df
```

The address of African Alliance Kenya Asset Management Limited doesnt seem to be valid, lets populate it with P.O. Box 27639 Nairobi 00506

²Approved Fund Managers by CMA. <https://www.cma.or.ke/licensees-market-players/>

```
fund_managers_df.loc[fund_managers_df['LICENCE NO.'] == 165, 'ADDRESS'] = 'P.O. Box 27639 Na  
fund_managers_df
```

Add Columns

- Location Coordinates (and google map)
- Headquter location/address/country
- Launch Date
- Risk Profile
- Trustee
- Custodian
- Auditors
- Minimum Investment
- Minimum Additional Investment
- Initial Fee
- Annual Management Fee
- Distribution
- Asset Under Management/Market share
- Advertised Rate [Gross, Net]
- Duration to withdraw
- Security - joint account verification/validation
- has online portal
- Withdraw charges
- Contacts

Getting the Perormance

According to Capital Markets Authority, fund managers are required to publish their yields daily in a reputable newspaper. What this means accessing hostorical records requires a significant investment in time to collect, aggregate and validate the published yields. That not withstanding, two very important questions become obvious: - what is a reputbale newspaper? - what yield do they publish, gross or net? - What picture does an anualized daily rate paint? - Is interest deffered or carried forward (reporting a convervative figure and retaining the rest for rainy day to preserve a picture of good performance)?

Luckily, one of the fund managers, one that has recently found its self in the courts far more often than it would have wished, does exelent investment and market research. It has a good-enough aggregate of fund managers and their performance weekly and monthly. We are going to crawl their data for analysis, and analyze it for manipulation.If the data comeout clean, we will analyze the trend of fund managers using the data.

Cytonn Research

We are going to crawl the data from cytonn research, <https://cytonn.com/researches/categories/1>

screen shots

Lests start with a view of weekly reports

```
await web_screenshot_async(  
    "https://cytonn.com/researches/categories/1",  
    width=1000)
```

Here is the latest report

```
await web_screenshot_async(  
    "https://cytonnreport.com/research/progress-of-retirement",  
    width=1000)
```

Instead of directly crawling HTML from <https://cytonn.com/researches/> page, we can instead crawl JSON from <https://cytonnreport.com/research> page, using the link <https://cytonnreport.com/get/allreports>.

```
async def get_all_cytonn_reports(per_page_count: int = 10):  
    page, browser = await get_browser_page_async()  
    reports_url = "https://cytonnreport.com/get/allreports"  
    reports_headers: dict = None  
    reports_method: str = None  
    async def catch_request(request: Request):  
        nonlocal reports_headers  
        nonlocal reports_method  
        if request.url == reports_url:  
            reports_headers = request.headers.copy()  
            reports_method = request.method  
            await request.continue_()  
        else:  
            await request.continue_()  
    async def get_cytonn_reports(current_page: int):  
        js_fetch_fn = f'''  
            async () => {{  
                const response = await fetch(  
                    "{reports_url}",  
                    {{  

```

```

        "headers": {json.dumps(reports_headers)},
        "referrer": "https://cytonnreport.com/research",
        "referrerPolicy": "no-referrer-when-downgrade",
        "body": {json.dumps(json.dumps(
            {
                "pagination": {
                    "per_page": per_page_count,
                    'current_page': current_page
                }
            }
        )))},
        "method": "{reports_method}",
        "mode": "cors",
        "credentials": "include"
    }));
    const json = await response.json();
    return json;
}}
'''
response_json = await page.evaluate(js_fetch_fn)
return response_json
# Enable request interception
await page.setRequestInterception(True)
# Attach the request handler
page.on('request', lambda request: asyncio.ensure_future(catch_request(request)))
# Navigate to the desired URL
await page.goto("https://cytonnreport.com/research")
while not reports_headers:
    await asyncio.sleep(1)
current_page = 1
all_reports = []
pbar: tqdm = None
while True:
    reports_response = await get_cytonn_reports(current_page)
    reports = reports_response['data'] if reports_response else []
    if len(reports) > 0:
        total = reports_response['total']
        pbar = pbar or tqdm(total=total)
        pbar.update(len(reports))
        all_reports.extend(reports)
        last_page = reports_response['last_page']
        if last_page == current_page:
            break

```

```

        current_page += 1
    else:
        break
    await browser.close()
    if pbar:
        pbar.close()
    return all_reports

all_cytonn_reports = await get_all_cytonn_reports()
print(f'There are {len(all_cytonn_reports)} reports')

# https://charanhu.medium.com/converting-pandas-dataframe-into-a-dataset-and-pushing-to-hugg
all_cytonn_reports_df = pd.DataFrame(all_cytonn_reports)
with pd.option_context(
    'display.max_columns', None,
    'display.max_colwidth', 100):
    display(all_cytonn_reports_df)

# all_cytonn_reports_df[['researchdate', 'created_at', 'updated_at', 'deleted_at', 'date']]

import re

def match_fn(x: str, y:str) -> bool:
    return \
        x.strip().lower() == y.strip().lower() or\
        re.sub(r'\s+', ' ', x.strip().lower()).replace(" ", "-") == re.sub(r'\s+', ' ', y.st

def get_table_df(html: str, terms: list[str] = ['Rank', 'Fund Manager', 'Effective Annual Ra
    first_report = BeautifulSoup(html, "html.parser")
    for tag in first_report.find_all(True):
        tag.attrs = {} # remove tage such as colspan and rowspan
    tables: list[Tag] = [table for table in first_report.find_all('table')]
    for table in tables:
        header_tr_s: list[Tag] = table.select('thead tr')
        is_match = False
        for header_tr in header_tr_s:
            header_td_s: list[Tag] = header_tr.find_all('td')
            is_match_new = len(header_td_s) == len(terms) and all([match_fn(header_td.get_te
            if not is_match_new:
                header_tr.extract()
            is_match = is_match or is_match_new
        if is_match:

```

```

        try:
            table_dfs = pd.read_html(io.StringIO(str(table)))
            if len(table_dfs) > 0:
                return table_dfs[0]
        except:
            continue
    return None

def get_table_with_df_by_topics(topics: list[dict], *args):
    all_topic_bodies = ' '.join([i['body'] for i in topics])
    table = get_table_df(all_topic_bodies, *args)
    return table

get_table_with_df_by_topics(all_cytonn_reports_df.loc[1, 'topics'])

def is_dataframe_empty(df: pd.DataFrame | None) -> bool:
    return df is not None and not df.empty

topics_columns_list = [
    ['Rank', 'Fund Manager', 'Effective Annual Rate'],
    ['Rank', 'Fund Manager', 'Effective Annual'],
    ['Rank', 'Fund Manager', 'Daily Yield', 'Effective Annual Rate'],
    ['(Kshs mn)', 'Market Share', '(Kshs mn)', 'Market Share', 'FY'2023 -Q1'2024"],
    ['no.', 'fund-managers', 'q1'2020-aum(kshs-mns)', 'q1'2020market-share', 'q2'2020-aum(kshs-mns)'],
    ['no.', 'fund-managers', "fy'2019-aum(kshs-mns)", 'q1'2020-aum(kshs-mns)', "aum-growth*fy'2019-aum(kshs-mns)"],
    ['no.', 'fund-managers', "fy'2018-aum-(kshs-mns)", "h1'2019-aum-(kshs-mns)", "aum-h1'2019-aum(kshs-mns)"],
    ['no.', 'money-market-fund', '2018-average-effective-annual-yield-p.a.'],
    ['no.', 'fund-managers', 'q2'2020-aum', 'q2'2020', 'q3'2020-aum', 'q3'2020', 'aum-growth*fy'2019-aum(kshs-mns)'],
    ['rank', 'money-market-funds', 'effective-annual-rate-(average-q3'2020)'],
    ['no.', 'fund-managers', "fy'2018-aum(kshs-mns)", "fy'2019-aum(kshs-mns)", "aum-growthfy'2019-aum(kshs-mns)"],
    ['no.', 'fund-managers', "fy'2018-money-market-fund(kshs-mns)", "fy'2019-money-market-fund(kshs-mns)"],
    ['rank', 'money-market-funds', 'effective-annual-rate-(average-fy'2019)']
]

# some pages have more than one table, ge: https://cytonnreport.com/research/unit-trust-fund
def format_row(row) -> bool:
    for topics_columns in topics_columns_list:
        df = get_table_with_df_by_topics(row['topics'], topics_columns)
        if is_dataframe_empty(df):
            return True
    return False

```



```
tqdm.pandas(desc="by_topics")
all_cytonn_reports_df['by_topics'] = all_cytonn_reports_df.progress_apply(format_row, axis=1,
```

```
len(all_cytonn_reports_df[all_cytonn_reports_df['by_topics'] == True])
```

```
cant_get_by_topic_df = all_cytonn_reports_df[all_cytonn_reports_df['by_topics'] == False].re
cant_get_by_topic_df
```

```
no_unit_trust_tables = [436]
```

```
import re

def regex_contains(text, substring):
    pattern = re.escape(substring)
    return bool(re.search(pattern, text, flags=re.IGNORECASE))

keywords = [
    "Approved Collective Investment Schemes",
    # "Collective Investment Schemes",
    # "Unit Trust Fund"
]

check_indexes = []

for index, row in tqdm(cant_get_by_topic_df.iterrows()):
    string_value = ' '.join(str(value) for value in row)
    if any([regex_contains(string_value, i) for i in keywords]):
        check_indexes.append(index)
check_indexes
```

```
cant_get_by_topic_df.loc[57, 'url']
```

```
import webbrowser

webbrowser.get("/usr/bin/google-chrome %s")
for check_index in check_indexes:
    url = str(cant_get_by_topic_df.loc[check_index, 'url'])
    webbrowser.open(url)
```

```

sdfdsfdsf = [
    "Rank",

    "Money Market Funds",

    "Effective Annual Rate (Average FY'2019)"
]

sdfdsfdsf = [re.sub(r'\s+', ' ', i.strip().lower()).replace(" ", "-") for i in sdfdsfdsf]
sdfdsfdsf

```

```

get_table_with_df_by_topics(
    cant_get_by_topic_df.loc[108, 'topics'],
    sdfdsfdsf)

```

```

cant_get_by_topic_df.loc[31, 'topics']

```

Disclaimer: The analysis, conclusions, and information provided in this publication represent the views and interpretations of the authors and researchers involved. While ToKnow.ai strives to provide accurate and reliable information, the specific details, data, and findings are not warranted or guaranteed. This analysis is intended solely for general informational purposes and should not be construed as financial, legal or other professional advice, recommendations, solicitations, or any other call to action. **Read more:** [/terms-of-service](#)